SQL

Mot de passe : **Pass123 !**

MySQL : Une base de données

# SGBD / SGBDR

## Défintions

Une base de donnée = est un ensemble de donnée qui ont été stockés sur un support info qui ont été organisé et structuré de façon à pouvoir modifier leurs données.

Il faut des outils pour administrer cette base de données, la modifier et pouvoir communiquer avec.

Il y a plusieurs types de bases de données : les bases de données relationnelles = MySQL

Il y a aussi des bases de données propriétaires

Des BdD NOSQL, qui ne sont pas relationnelles

Il y a 20 ans, il y avait des bases de données hiérarchiques (on n’en trouve plus).

Des ex : MySQL, Oracles, Access

Système de gestion de base de données : SGBDR

Va nous permettre de modifier, gérer nos bases de données = du GRUND (?)

Le R de SGBDR = relationnelle = Une théorie que va utiliser la base de données, toutes les données seront liés par une relation mathématiques

Un schéma : dedans il y a des tables de bases de données, et dans ces tables il y a des lignes = tuc = enregistrement, et sur chaque ligne on trouvera la mm structure qui sont des colonnes

Le langage SQL va nous permettre de faire la manipulation de nos données.

La ddl : des éléments qui nous permettent de modifier les data,

La dcl : permet de gerer les autorisation au niveau de notre base de données

Les TCL = les transactions de contrôle, gère les transactions en SQL

On a un modèle SQL serveur,

MySqld.exe : = la partie serveur

La partie cliente : se connecte à cette base de données :

mySql.exe

workbench = un client graphique

Les programmes sont aussi des clients de la base de donnée

Les clients communiquent grâce au langage SQL

Un serveur = une entité qui va tendre des requêtes et qui va émettre des réponses par rapport à des demandes 🡺 il possède les donnée

Un client est une entité qui va se connecter au serveur faire une demande et avoir une réponse. C’est le consommateur de la donnée à distance.

Un port = une entrée sur l’ordi

L’ordi a : IP = adresse logique d’un ordi (internet protocol)

La notion de port ~ boites postales

Chaque port correspond à une personne = un programme sur l’ordi

Qand on communique avec le serveur :

On lui communique l’adresse ou l’on veut aller et on lui dit que l’on veut communiquer avec mysql ,

Port + adresse IP = sockets

Toutes les machines peuvent s’auto identifier : avc l’adresse : 127.0.0.1 = qui signifie que l’on pointe sur la machine sur laquelle on travaille.

Workbench demande l’adresse IP de la machine et le socket pour pouvoir se connecter

SGBD : Système de gestion de base de donnéees

= un logiciel qui permet de stocker des info dans une base de donnes. Un tel système permet de ilre ecrire modifier trier transformer ou mm imprimer les donées qui sont contenus dans la base de données.

Il y a plusieurs catégories :

-les systemes propriétaires = du relationnelle

- des systemes libres

Ceux qui sont orienté objet

* Les embarqués
* Les noSQL
* Les autrs systemes (cf diapo)

Mise en œuvre (=implémentation) des bde données =

Les différentes catégories de données :

Deux questions :

* A quoi elles servent ?
* Ff

Les données de bases

Les données en mouvements

Le modèle relationnel

Assertionnel : langage ou on fait des tests (conditions etc)

Logique ensembliste : théorie des ensembles

Domaine = enesmble de valeurs caractérisé par un nom

Cardinal = nombre d’élements dans le domaine

Entre braket = un ensemble, les nom sont séparé par des virgules

Le produit cartésion P entre plusieurs domaines = l’ensemble des n-uplets ou chaque di est un element de D

= produit développé

Relation = un sous ensemble (..) on fait pas le produit cartésien complet, on prent que une partie de la relation

Attribut : colonne d’une relation caractéisée par un nom

Degré = nombre attributs d’une relation

Nombre taux

1

2

3

4

6

1.6

20

Attribut =2

Selet nombre, taux from –

La normalisation du schema relationnel

Premiere forme normale : respecte l’unicité

Deuxieme forme normale :

Schemas = une base de donné

Query =

#premiere requete de type dml

SELECT 'Hello, the world!' as 'Hello' ;

#String = en simple '', as permet de changer le nom de l'attrbut

Ctrl + entre = eclair

#premiere requete de type dml

SELECT (4\*6) as 'Calcul' ;

#String = en simple '', as permet de changer le nom de l'attrbut

On peut rajouter une tabulation \t et un saut de ligne \n

l\'ma = caractère d’échanpemet

SELECT 'Voici quelques résultats' as ' ' ,(2\*4) as '2x4', (4%2) as '4%2' , (3\*3) as '3x3' ;

# creation d'une base

CREATE DATABASE eleve ;

DROP DATABASE eleve ; #pour supprimer la base

# creation d'une base

#CREATE DATABASE eleve ;

#creation d'une première table :

#use eleve ; #on indique qu'on veut etre sur la vase eleve ou eleve.identite

CREATE TABLE eleve.identite ( #table = le domaine, et apres on a des colonnes

id INT PRIMARY KEY NOT NULL, #la colonne s'appelle ID, on lui donne le type, primary key = c'est l'identifiant du tupes, not null cette valeur ne peut etre nul

prenom varchar(255),

nom varchar(255),

information text

);

#int = un entier,

#char= chaine de caractere qui fait l'espace qu'on lui indique et varchar prend uniquement le nombre de char necessaire

#text = pour un texte

30/03/2017

My SQL possède différents moteur pour générer des tables.

# Les types de données

Pour créer une table, il faut indiquer le type pour savoir cb de mémoire il faut réserver:

## Les types numériques

* TINYINT = 1 octet ; de -128 à +127 (ils sont signées, le premier bit = le signe et le deuxième = le chiffre)
* SMALLINT = 2 octets (équivaut au char)
* MEDIUMINT = 3 octets
* INT = 4 octets (équivaut au int)
* BIGINT = 8 octets (équivaut au long)

On peut indiquer que la valeur sera insigné (Int UNSIGNED) : le type int non signé ira de 0 à 256 au lieu de -128 à 127. On réapproprie le bit devant qui servait au signe.

Les nombres décimaux

DOUBLE = 8 octets

NUMERIC

REAL

FLOAT = 4 octets

DECIMAL

// Numérique et décimal sont les mm, ils prennent deux paramètres : la précision et l’échelle.

Exemple :

Prix DECIMAL (3,2)

Le chiffre sera écrit : 400,45

// les chiffres entre parenthèse montre le nombre de paramètres/chiffre que l’on veut afficher

Double et real : ne prend pas de paramètres mais stocke les résultats avec plus de précisions. Double est plus grand et donc plus précis que real.

Numeric et decimal sont stockés sous forme de caractère, on a donc le nombre exacte alors que double et real non, ils sont stockés sous forme d’arrondi. !! à la précision.

## Les types alphanumériques

CHAR

VARCHAR = dynamique (on lui donne un max entre ())

Pour le texte

TINYTEXT

TEXT (en général est le plus utilisé, 90% des cas)

MEDIUMTEXT

LONGTEXT

// Stocker des images = se faisait avant mais c’est une mauvaise pratique. = à éviter au max

## Les dates et heure

DATE = stocke une date AAAA-MM-JJ

TIME = stocke une date et une heure HH:MM:SS

DATETIME = les deux AAAA-MM-JJ HH:MM:SS

YEAR = AAAA

TIMESTAMP = donne le nombre de seconde depuis le 1er janvier 1970 //très utilisé en info, il est plus facile à manipuler quand on veut faire des manipulations avec les dates

Hier on a vu :

//comment mettre un élément non null

// Comment mettre une clef primaire

EXO : Créer une base de donnée « Topzik » qui contiendra une table artiste où il y aura le nom, le prénom, une date de naissance, une date de décès si il est mort et autrement = NULL, et informations

CREATE DATABASE TopZik

CREATE TABLE topzik.Artiste (

Nom VARCHAR (255) PRIMARY KEY NOT NULL,

Prenom VARCHAR(255),

Date\_naissance DATE,

Date\_deces DATE,

information TEXT

);

Comment modifier une table :

On utilise : ALTER TABLE

ADD pour ajouter

DROP pour supprimer

CHANGE pour modifier ou MODIFY

Ajouter une colonne instrument :

*ALTER TABLE topzik.artiste ADD COLUMN instrument VARCHAR (255);*

Changer le type d’une variable ;

ALTER table topzik.artiste modify column nom varchar (55);

On va avoir une DML particulière. On a deux méthodes :

Insert INTO ARTIST

Values (‘jjhbjhb’,’jij’,’ijij’,NULL,’hh’) ;

Insert into artiste (‘fsf’,’mmmm’)

Values (mmmmmmmm)

Autre manière :

Injectez

Load DATA LOCAL inFILE ‘jgjkjg.CSV’

INTO TABLE nomtable

FIELDS TERMINATED BY ‘ ;’ ENLOAD’ »’

Line by ‘\n’

(nom,prenom,date,dgdf)

Ds le fichier excel : « nom » « prenom »



*load data local infile 'C:/Users/ib/Desktop/Classeur1.csv'*

*into table topzik.artiste*

*fields terminated by ';'*

*enclosed by''*

*lines terminated by '\n' ;*

Pour afficher tout le tableau :

Select \* from topzik.artiste;

Pour sélectionner des donner :

**SELECT**

**SELECT \* = pas une bonne pratique**

***Select 'Voici l\'artiste', nom, prenom from topzik.artiste;***

La condition

*Select nom de colonne from table where nom de colonne et valeur qu’on attend*

*select \* from topzik.artiste where prenom ='fanta';*

*On a des opérateurs arithmétiques :*

*< > = ==*

*On a aussi les OR, on peut combiner plusieurs operateurs*

*select \* from topzik.artiste where date\_naissance < '1996-05-05' and date\_naissance > '1990-01-01' ;*

Afficher dans l’ordre décroissant :

*select \* from topzik.artiste ORDER BY Date\_naissance desc;*

*Where nom like « bern% » = tout les noms commencant par bern*

**CARACTERE JOCKER : %**

*Si on veut afficher un 60% il faudra l’échapper : 60\%%*

*Colonne notlike « gfg% » = tt ce qui est pas %*

Like binary ‘F%’ 🡺 il cherchera que les F majuscule

Like ‘F%’ 🡺 il va chercher tous les F, min et majuscules

select \* from topzik.artiste where prenom like 'M%' ;

**Faire des recherches dans des intervalles :**

#select \* from topzik.artiste where date\_naissance < '1996-05-05' and date\_naissance > '1990-01-01' ;

Ou bien

**Le mot clef BETWEEN**

Where colonne in (‘mmm’,’mmm’,’mmm’) ;

// créer une deuxieme table liste avec des prenoms et faire une requete qui va retourner toutes les personnes qui ont un prenoms dans la liste donnée 🡺 hyper consommateurs

Poour parcourir deux tableaux et trouver les prénoms communs aux deux :

use topzik ;

Select distinct prenom from artiste where prenom in (select prenom from personnage) ;

= tres consommateurs

Comment supprimer des infos :

Delete from <table> where colonne = ‘mmm’ ;

UPDATE <table> set colonne='mmmmm' where nom=mmmm;

// si pas de where on va tt changer

31/03/2017

## Les clefs

Cohérence, unicité et identifiant = les trois règles à respecter.

La clef primaire permet de garder la notion d’unicité et de cohérence. Comment créer un primary key :

Colonne int unsigned PRIMARY KEY **AUTO\_INCREMENT** // génère automatiquement une cley sur mySQL !!! La génération automatique est différente sur oracles. Unsigned pour avoir plus de possibilités (de 0 à 256).

#use eleve

/\*

CREATE TABLE eleve.test (

id INT UNSIGNED PRIMARY KEY AUTO\_INCREMENT,

Nom VARCHAR (255) ,

Prenom VARCHAR(255)

);

insert into eleve.test (nom, prenom)

values

('Smajovic', 'Mirela'),

('nom2', 'prenom2'),

('nom3','prenom2'),

('nom4', 'prenom4'),

('nom5','prenom5');

\*/

select \* from eleve.test ;

#use eleve

/\*

CREATE TABLE eleve.test (

id INT UNSIGNED PRIMARY KEY AUTO\_INCREMENT,

Nom VARCHAR (255) ,

Prenom VARCHAR(255)

);

insert into eleve.test (nom, prenom)

values

('Smajovic', 'Mirela'),

('nom2', 'prenom2'),

('nom3','prenom2'),

('nom4', 'prenom4'),

('nom5','prenom5');

\*/

select \* from eleve.test ;

# Les clefs étrangères

Permet la cohérence :

Methode 1 pdt la creation :

Create table (

Mmm ,

Mmm,

Contraint

Foreign key (civilité) references (civilité (id)

)

Methode 2 pendant une modification :

ALTER TABLE Client = pour modifier une table

ADD constraint fffff foreign key (--)

Si on veut la supprimer :

Drop foreign key fk………

Faire une base contact avec deux tables : client et civilités qui applique cette containte : essayer de jouer avec les différentes commandes : creation de contrainte, suppressio, modification. Enrengistrer une dizaine de contacts avec les civilités

Les jointures :

Select \* from client

Inner join civilite

On (//dont la condition sera) clien.civiliteID = civilite.ID

Tuples

CREATE DATABASE Contact2;

Use contact2;

CREATE TABLE civilite (

id INT UNSIGNED PRIMARY KEY AUTO\_INCREMENT,

Civilite VARCHAR(255)

);

INSERT INTO civilite (civilite)

VALUES

('Mme'),

('Mlle'),

('Mr'),

('Mme et M');

CREATE TABLE contact2.client1 (

id2 INT UNSIGNED PRIMARY KEY NOT NULL AUTO\_INCREMENT,

Nom VARCHAR(255),

Prenom VARCHAR (255),

id\_client int unsigned not null,

constraint fk\_client\_civilite foreign key (id\_client) references contact2.civilite(id)

);

#alter table client drop foreing key fk\_civiliteID;

INSERT INTO contact2.client1 (nom, prenom, id\_client)

VALUES

('nomgarçon', 'prenomgarçon',3),

('nomgarçon2', 'prenomgarçon2',3),

('nomfllette2', 'prenomfillette2',2),

('nomfemme', 'prenomfemme',1),

('nomfllette3', 'prenomfillette3',2);

select civilite.civilite, client1.nom , client1.prenom from client1

inner join civilite on client1.id\_client = civilite.id ;

select civilite.civilite, client1.nom , client1.prenom from client1

inner join civilite on client1.id\_client = civilite.id

where civilite.civilite ='mlle';

inner join = fait la liaison entre a et b 🡺 jonctions internes

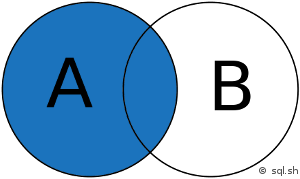
jocntions externes :

tout a et et b uniquement ce qui est partagé avec a

from client = table gauche LEFT JOIN table droite

select \* from artiste

left join instruments on artiste.id = instruments.id2;



On a la mm chose du côté droit :

Avec Right join